Динамические структуры данных (пример)

Лекция 10 (окончание)

Задание

Составить программу, формирующую сведения об отправлении поездов.

Каждый поезд описывается с помощью структуры, содержащей следующие поля:

– номер поезда (ключ списка);

– пункт назначения;

– время отправления.

Данные хранятся в виде двунаправленного линейного списка.

Задание

Программа должна реализовывать следующие операции:

– формирование меню для выбора действия пользователем;

– ввод данных в список, отсортированный по ключу;

– вывод данных из списка в виде таблицы;

– поиск элемента в списке по различным критериям (номеру поезда, имени пункта назначения, времени отправления);

– удаление элемента из списка по заданному ключу.

Текст программы

```
#include "stdafx.h"
#include "stdio.h"
#include "locale.h"
#include "conio.h"
#include <iostream>
using namespace std;

//Структура для формирования списка
struct Train {
    int train_num;     //Номер поезда
    char p_nazn[20];//Имя пункта назначения
    char v_otpr[10]; //Время отправления
    Train *prev;       //Указатель на предыдущий элемент
    Train *next;       //Указатель на последующий элемент
};
```

```
Train *pbeg;     //Указатель на начало списка
Train *pend;     //Указатель на конец списка

//Список функций программы
void menu();     //Функция прорисовки меню

//Функция формирования первого элемента списка
Train* first(int t_num, char* p_nazn, char* v_otpr);

//Функция вставки элемента в упорядоченный список
void add_sort(Train **pbeg, Train **pend, int t_num, char* p_nazn,
    char* v_otpr);

//Функция поиска элемента по ключу (номер поезда)
Train* find(Train* const pbeg, int t_num);

//Функция поиска элемента по имени пункта назначения
bool find_punkt(Train* const pbeg, char* p_nazn);
```

```
//Функция поиска элемента по времени отправления
bool find_time(Train* const pbeg, char* v_otpr);

//Функция удаления элемента из списка по заданному ключу
bool remove(Train **pbeg, Train **pend, int key);

void vvod();     //Функция ввода данных в список

void vivod();    //Функция вывода элементов списка на экран

void search();   //Функция поиска элементов в списке

void del();      //Функция удаления элемента из списка

void hline();          //Функция прорисовки горизонтальной
                       // линии
```

```
//Главная функция программы
int _tmain(int argc, _TCHAR* argv[])
{
   setlocale(LC_ALL,"");
   pbeg = 0;   //Инициализация указателя на первый
         //элемент списка
   pend = 0;   //Инициализация указателя на
    //последний элемент списка
   int num=0;//Переменная для хранения номера
    //выбранного пункта меню
```

```
//Цикл обработки действий пользователя
    while (num!=5)
    {
     menu();
     cin>>num;
     switch(num){
           case 1: vvod(); break;
           case 2: vivod(); break;
           case 3: search(); break;
           case 4: del(); break;
           case 5: break;
           default: cout<<"Неверный вариант выбора";
     }

    }
    return 0;
}
```

```
//Функция прорисовки меню
void menu()
{
   cout<<"\n  МЕНЮ \n";
   cout<<"1. Ввод данных\n";
   cout<<"2. Вывод данных\n";
   cout<<"3. Поиск\n";
   cout<<"4. Удаление\n";
   cout<<"5. Выход\n";
   cout<<"\nВведите номер пункта меню для
   дальнейшей работы\n";
}
```

```
//Функция формирования первого элемента списка
Train* first(int t_num, char* p_nazn, char* v_otpr)
{
   //Создание нового элемента
   Train *pv = new Train;
   //Заполнение полей данных
   pv->train_num = t_num;
   strcpy(pv->p_nazn, p_nazn);
   strcpy(pv->v_otpr, v_otpr);
   //Инициализация указателей на соседние элементы
   //списка
   pv->prev = 0;
   pv->next = 0;
   return pv;
}
```

```
void add_sort(Train **pbeg, Train **pend,
  int t_num, char* p_nazn, char* v_otpr)
{
  //Создание нового элемента
  Train *pv = new Train;
  //Заполнение полей данных
  pv->train_num = t_num;
  strcpy(pv->p_nazn, p_nazn);
  strcpy(pv->v_otpr, v_otpr);
  Train *pt = *pbeg;
```

```
while (pt){                    //Просмотр списка
    if (t_num < pt->train_num)
    {
        pv->next = pt;
        if (pt == *pbeg)  //Вставка в начало списка
        {
            pv->prev = 0;
            *pbeg = pv;
        }
        else                  //Вставка в середину списка
        {
            (pt->prev)->next = pv;
            pv->prev = pt->prev;
        }
```

```
pt->prev = pv;
        return;
    }
     pt = pt->next;
    }
    pv->next = 0;        //Вставка в конец списка
    pv->prev = *pend;
    (*pend)->next = pv;
    *pend = pv;
}
```

```
//Функция поиска элемента по номеру
   поезда
Train* find(Train *const pbeg, int t_num)
{
   Train *pv = pbeg;
   while (pv)
   {
    if (pv->train_num == t_num)
         break;
    pv = pv->next;
   }
   return pv;
}
```

```
//Функция поиска элемента по имени пункта назначения
bool find_punkt(Train* const pbeg, char* p_nazn)
{
    bool find=false;
    Train *pv = pbeg;
    while (pv) //Просмотр списка
    {
     if (strcmp(pv->p_nazn, p_nazn)==0)
     {
            find=true;
            //Вывод на экран найденного элемента
            printf("%5d %20s %10s\n", pv->train_num, pv-p_nazn,
    pv->v_otpr);
     }
     pv = pv->next;
    }
    return find;
}
```

```
//Функция поиска элемента по времени отправления
bool find_time(Train* const pbeg, char* v_otpr)
{
    bool find=false;
    Train *pv = pbeg;
    while (pv) //Просмотр списка
    {
     if (strcmp(pv->v_otpr, v_otpr)==0)
     {
            find=true;
            //Вывод на экран найденного элемента
            printf("%5d %20s %10s\n", pv->train_num, pv>p_nazn,
    pv->v_otpr);
     }
     pv = pv->next;
    }
    return find;
}
```

```
//Функция удаления элемента из списка
bool remove(Train **pbeg, Train **pend, int
   key)
{
   if (Train *pkey = find(*pbeg, key))
   {
    //Если удаляется первый элемент из списка
    if (pkey == *pbeg)
    {
        *pbeg = (*pbeg)->next;
        if (*pbeg) (*pbeg)->prev = 0;
    }
```

```
else
        //Если удаляется последний элемент из списка
        if (pkey == *pend)
        {
            *pend = (*pend)->prev;
            (*pend)->next = 0;
        }
        //Если удаляется элемент из середины списка
        else
        {
            (pkey->prev)->next = pkey->next;
            (pkey->next)->prev = pkey->prev;
        }
        delete pkey;
        return true;
    }
    return false;
}
```

```
//Функция ввода данных
void vvod()
{
  int n;
  int t_num;
  char p_nazn[20];
  char v_otpr[10];
  //Ввод количества добавляемых записей
  cout<<"Введите кол-во добавляемых
  записей в список = ";
  cin>>n;
```

```
//Цикл для ввода полей структуры
    for (int i=0; i<n; i++)
    {
     cout<<"\nВведите номер поезда\n";
     cin>>t_num;
     cout<<"Введите имя пункта назначения\n";
     fflush(stdin);
     gets(p_nazn);
     cout<<"Введите время отправления\n";
     fflush(stdin);
     gets(v_otpr);
```

```
//Если в списке есть элементы - добавляется новый
// элемент с сортировкой списка
    if (pbeg)
        add_sort(&pbeg, &pend, t_num, p_nazn, v_otpr);
    //Если список пуст - формируется первый
    // элемент списка
    else
    {
        pbeg = first(t_num, p_nazn, v_otpr);
        pend = pbeg;
    }
   }

}
```

```
//Функция вывода списка на экран в виде
   таблицы
void vivod()
{
   if (pbeg) //Если список не пуст - вывод на
   экран
    {
     //Прорисовка шапки таблицы
     cout<<"\n                Список поездов\n";
     hline();
     printf("| Номер |   Пункт назначения   |
   Время отп. |\n");
     hline();
```

```
//Вывод элементов списка на экран
    Train *pv = pbeg;
    while (pv)
    {
        printf("| %5d | %20s | %10s |\n",
pv->train_num, pv->p_nazn, pv->v_otpr);
        pv = pv->next;
    }
    //Отчеркивание нижней гранцы таблицы
    hline();
   }
   else      //Если список пуст - вывод сообщения
    cout<<"\nСписок пуст!\n";
}
```

```
//Функция поиска элементов в списке
void search()
{
   if (pbeg)
   {
   int k_poisk;
   //Прорисовка меню выбора критерия поиска
   cout<<"\nВыберите критерий поиска\n";
   cout<<"1. Поиск по номеру поезда\n";
   cout<<"2. Поиск по имени пункта назначения\n";
   cout<<"3. Поиск по времени отправления\n";
   cout<<"\nВведите номер критерия ";
   cin>>k_poisk;
```

```
//Обработка выбора критерия пользователем
    switch (k_poisk)
    {
     case 1: //Поиск по номеру поезда
            int t_num;
            cout<<"\nВведите номер поезда ";
            cin>>t_num;
            if (Train *pv = find(pbeg, t_num))
            {
                  cout<<"\nРезультат поиска:\n";
                  printf("%5d %20s %10s\n",
pv->train_num, pv->p_nazn, pv->v_otpr);
            }
            else
                  cout<<"Поезд с таким номером не найден\n";
            break;
```

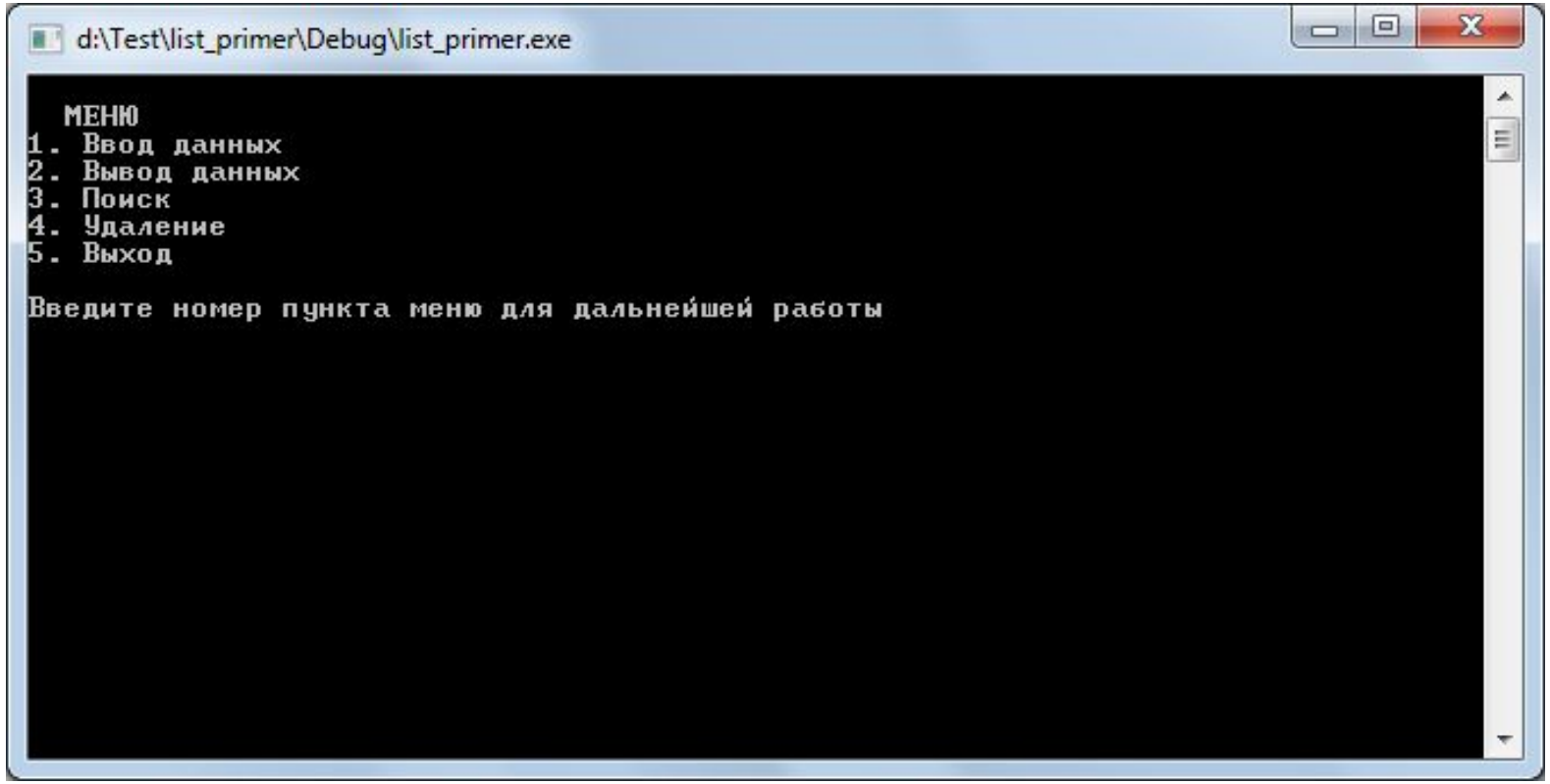
```
case 2:   //Поиск по имени пункта назначения
        char p_nazn[20];
        cout<<"\nВведите имя пункта
  назначения\n";
        fflush(stdin);
        gets(p_nazn);
        cout<<"\nРезультат поиска:\n";
        if (!find_punkt(pbeg, p_nazn))
            cout<<"Поезда с таким пунктом
  назначения не найдены\n";
        break;
```

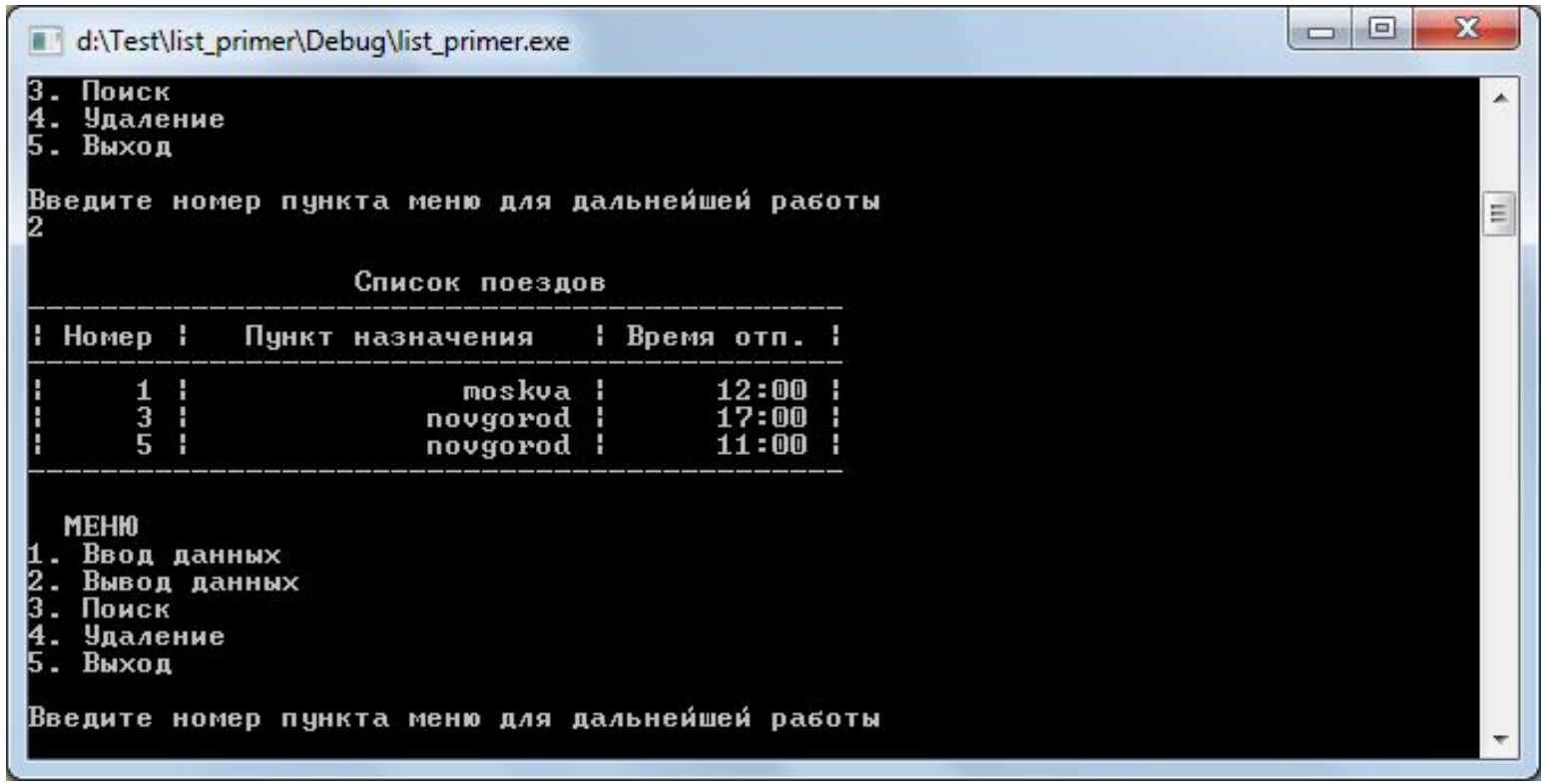
```
case 3: //Поиск по времени отправления
        char v_otpr[10];
        cout<<"\nВведите время отправления\n";
        fflush(stdin);
        gets(v_otpr);
        cout<<"\nРезультат поиска:\n";
        if (!find_time(pbeg, v_otpr))
            cout<<"Поезда с таким временем отправления не
   найдены\n";
        break;
    default: cout<<"Неверный вариант выбора\n";
   }
   }
   else
    cout<<"\nСписок пуст!\n";

}
```

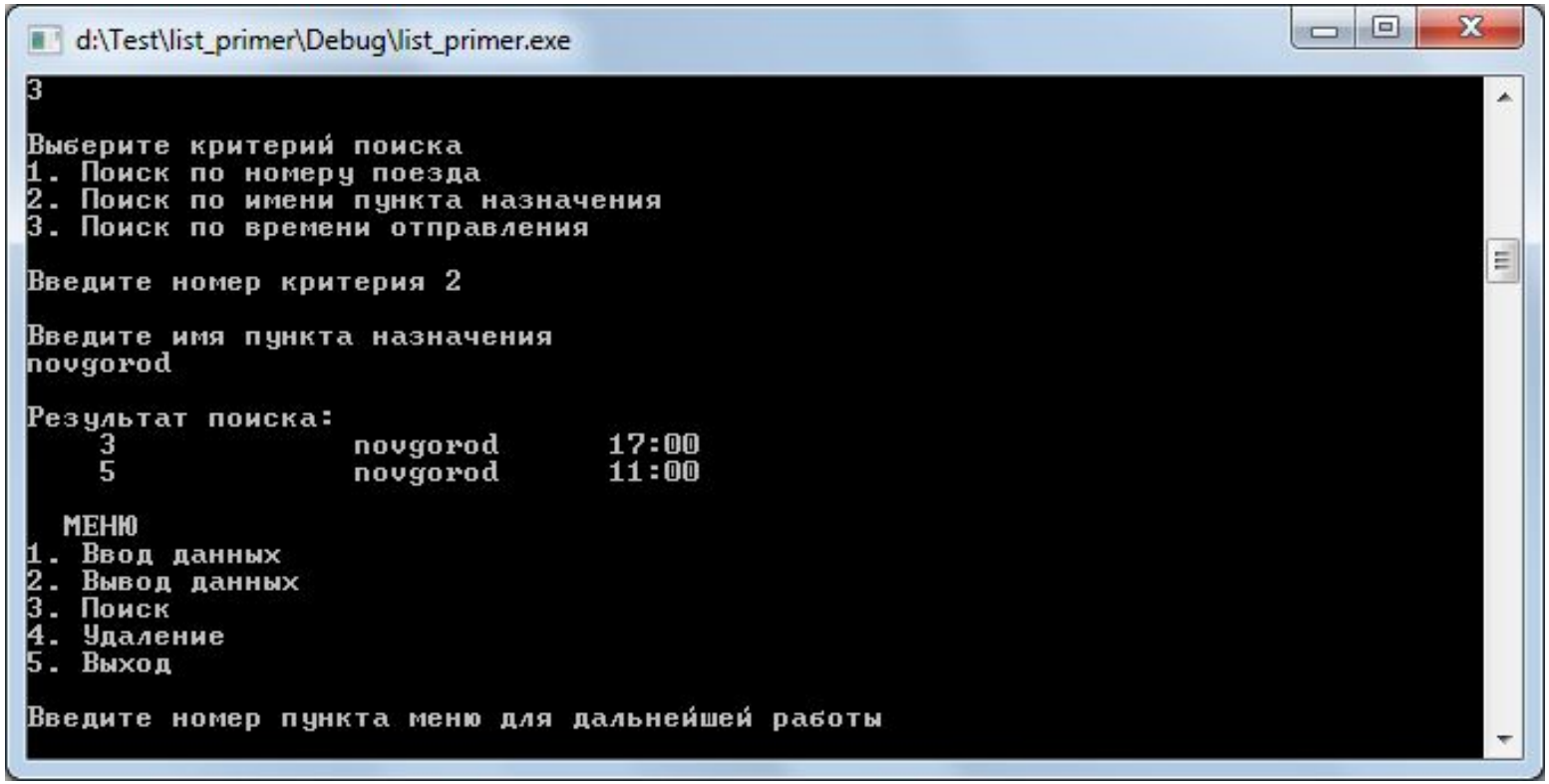
```
//Функция удаления элемента из списка
void del()
{
    if (pbeg)
    {
    int t_num;
    //Ввод номера поезда для поиска и удаления
    //соответствующей записи
    cout<<"Введите номер поезда ";
    cin>>t_num;
    if (remove(&pbeg, &pend, t_num))
     cout<<"\nЗапись удалена\n";
    else
     cout<<"\nЗапись с таким номером не найдена\n";
    }
    else
     cout<<"\nСписок пуст!\n";
}
```

```
//Функция для проведения горизонтальной
// линии в таблице
void hline()
{
   const int m=45;
   for (int i=0; i<m; i++)
    printf("-");
   printf("\n");
}
```

Результат работы




d:\Test\list_primer\Debug\list_primer.exe
3. Поиск
4. Удаление
5. Выход
Введите номер пункта меню для дальнейшей работы
2
Список поездов
Номер	Пункт назначения	Время отп.
1	moskva	12:00
3	novgorod	17:00
5	novgorod	11:00
МЕНЮ
1. Ввод данных
2. Вывод данных
3. Поиск
4. Удаление
5. Выход
Введите номер пункта меню для дальнейшей работы


d:\Test\list_primer\Debug\list_primer.exe
3
Выберите критерий поиска
1. Поиск по номеру поезда
2. Поиск по имени пункта назначения
3. Поиск по времени отправления
Введите номер критерия 2
Введите имя пункта назначения
novgorod
Результат поиска:
3 novgorod 17:00
5 novgorod 11:00
МЕНЮ
1. Ввод данных
2. Вывод данных
3. Поиск
4. Удаление
5. Выход
Введите номер пункта меню для дальнейшей работы